Hexahedral Mesh Generation Constraints

Jason F. Shepherd[1] and Chris R. Johnson[2]

[1] Scientific Computing and Imaging Institute, Salt Lake City, UT
jfsheph@sci.utah.edu
[2] Scientific Computing and Imaging Institute, Salt Lake City, UT
crj@sci.utah.edu

For finite element analyses within highly elastic and plastic structural domains, hexahedral meshes have historically offered some benefits over tetrahedral finite element meshes in terms of reduced error, smaller element counts, and improved reliability. However, hexahedral finite element mesh generation continues to be difficult to perform and automate, with hexahedral mesh generation taking several orders of magnitude longer than current tetrahedral mesh generators to complete. Thus, developing a better understanding of the underlying constraints that make hexahedral meshing difficult could result in dramatic reductions in the amount of time necessary to prepare a hexahedral finite element model for analysis. In this paper, we present a survey of constraints associated with hexahedral meshes (i.e. the conditions that must be satisfied to produce a hexahedral mesh). In presenting our formulation of these constraints, we will utilize the dual of a hexahedral mesh. We also discuss how incorporation of these constraints into existing hexahedral mesh generation algorithms could be utilized to extend the class of geometries to which these algorithms apply. We also describe a list of open problems in hexahedral mesh generation and give some context for future efforts in addressing these problems.

1 INTRODUCTION

Numerical approximation methods, including finite element, finite difference, and finite volume methods, are mathematical techniques used to model various scientific and engineering phenomena for a wide variety of disciplines, including structural mechanics, dynamics, heat transfer, and fluid dynamics. Because of the flexibility of these approximation methods, the problems to which they can be applied is growing. They are currently being utilized in fields ranging in diversity from cellular microbiology and quantum chromodynamics to star and galaxy formation studies [29, 33, 30]).

An important requirement of the numerical approximation techniques above is the need to create a discrete decomposition of the model geometry into a 'mesh'. The meshes produced are used as input for computational simulation, as well as, the geometric basis for which many of the visualization results are displayed.

The most common types of elements utilized in numerical approximations are triangles or quadrilaterals in two-dimensions and tetrahedral or hexahedral elements in three-dimensions. To reduce the amount of time to prepare a model, automated meshing algorithms have been developed for creating triangular, quadrilateral, and tetrahedral meshes for a very generalized class of geometries. In the case of tetrahedral meshing, algorithms are available that can generate greater than 400 thousand tetrahedra per minute [27]. However, automated hexahedral mesh generation algorithms are available for a more limited class of geometries. Because of the limited class of geometries for which hexahedral meshes can be built, a significant amount of time in generating a hexahedral mesh is devoted to decomposing (cutting up) a model into pieces for which a known hexahedral mesh generation algorithm will succeed. The processing of geometry for creating a hexahedral mesh can take several months for a generalized model, whereas tetrahedral meshes can often be created in a matter of hours or days [68, 69].

In spite of the limited availability of an automated hexahedral mesh generation algorithm, hexahedral meshes are sometimes preferred over tetrahedral meshes in certain applications and situations for the following reasons:

1. Tetrahedral meshes typically require 4-10 times more elements than a hexahedral mesh to obtain the same level of accuracy [65, 12].
2. In some types of numerical approximations (i.e. high deformation structural finite element analysis with linear elements), tetrahedral elements will be mathematically 'stiffer' due to a reduced number of degrees of freedom associated with a tetrahedral element [2, 10]. This problem is also known as 'tet-locking'.

Hexahedral mesh generation can be difficult and time-consuming. In this paper, we focus on delineating the underlying criteria that must be satisfied (i.e. constraints) in order to produce a hexahedral mesh for a given geometric model. This paper is intended to survey existing methods and ideas in hexahedral mesh generation and provide some basis for future research. In the remainder of this paper, we give a brief overview of hexahedral mesh generation, specifically in relation to the dual representation of a hexahedral mesh. Then, we outline topologic, boundary, and quality constraints that must be satisfied to generate a hexahedral mesh. We also give background on several methods that may be utilized to satisfy specific hexahedral constraints to capture spatial features within a hexahedral mesh. We conclude the paper by highlighting how incorporation of additional constraint satisfaction methods into existing algorithms can dramatically extend the class of geometries to

which these algorithms apply. We also identify several open problems in hexahedral mesh generation, and give some context for how these solutions might be addressed in future research.

2 BACKGROUND

Numerous algorithms exist for producing hexahedral meshes [41]. However, no one algorithm is completely successful at generating provably correct and robust hexahedral meshes. The most basic form of a hexahedral mesh generation algorithm stems from the mesh of a cuboid (i.e. subdivisions of a single hexahedral element). For complex geometries, meshes can be obtained by decomposing the initial geometry into collection of cuboids. For geometries that are encountered frequently, common decompositions can be retained, or stored, as a 'primitive' decomposition, and when the common shape is encountered again, meshing can be nearly automatic [7].

2.1 The Dual of a Hexahedral Mesh

Because decomposition of a model into cuboids and/or primitive shapes can be tedious, complex, and difficult to automate, alternative algorithms have been sought that do not require geometric decomposition.

In late 1993, hexahedral mesh generation was posed as a topological problem by Thurston [61] and Murdoch [39, 40]. They realized that utilizing the dual subdivision of a hexahedral mesh yields a structure of internal surfaces (also known as 'sheets'), where the structure of the sheets dictates the structure of the hexahedral mesh (and vice versa) according to some specific topological requirements.

For a given quadrilateral mesh on a surface, the topologic structure of the dual of the mesh can be visualized by drawing a line segment across each quadrilateral connecting opposite edges. For each quadrilateral, there are two such line segments, one for each of the opposite pairs of edges on a quadrilateral element. The intersection point of these two segments (which has been termed a 'centroid') is dual to the quadrilateral itself. By iteratively placing the dual line segments for each quadrilateral in the surface mesh, it is soon realized that each of the line segments from a quadrilateral connect neighbor to neighbor until the line segments form either a closed curve, or the resulting curve has end points at the surface boundary. Each of these dual curves is called a 'chord' in the quadrilateral mesh. An example of a mesh (also known as the 'primal'), with its dual, is shown in Figure 1.

Several important items can be gleaned from the dual representation of the quadrilateral mesh:

1. The intersection point of two chords is known as a 'centroid', and a centroid is dual to a quadrilateral in primal space.

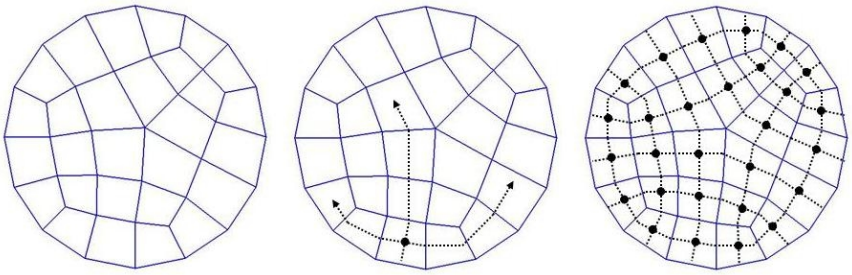

Fig. 1. A quadrilateral mesh on a circular disk (left). The dual of a quadrilateral mesh is created by line segments (chords) connecting opposite edges of an individual quad, and traversing all of the edges of all of the quads on the mesh (middle). The intersection of two chords is called a centroid and is dual to a quadrilateral. The complete dual is on the right.

2. Each chord represents a 'stack' of quadrilaterals in the primal mesh.
3. All chords must either have endpoints on the boundary of the surface, or they must form a closed curve within the boundary of the surface. (i.e. there is no chord with an endpoint in the interior of the surface.)
4. Chords cannot be tangent to other chords.
5. The 'size' of the primal mesh and the number of elements local to an area of the surface is a function of the density of chords and chord intersections relative to that locale on the surface.
6. As a consequence of observation three above, it can be shown that the parity of the edges around a surface admitting a quadrilateral mesh must be even.

Extending these observations of quadrilateral meshes to hexahedral meshes, we can formulate the dual of a hexahedral mesh. Since each hexahedral element will consists of three pairs of opposing quadrilaterals (similar to the two opposing edges for quadrilateral elements), we can draw line segments between the centers of each of the opposing faces of the hexahedra. We observe that, in similar fashion to the chords of a quadrilateral mesh, the chords within a hexahedral mesh define a stack of hexahedra. However, we also observe that these stacks now interact in two directions resulting in 'layers' of elements. A layer of elements corresponds to a surface in the dual space of the mesh. This dual surface has been referred to as a 'sheet' [34], an 'interior surface' [15], or as a 'twist plane' [40, 39]. For the purposes of this paper, we will utilize the term 'sheet' to apply to these dual surfaces. An example dual subdivision of a hexahedral mesh is shown in Figure 2.

Reviewing a valid hexahedral mesh with its' dual subdivision, the following observations can be made:

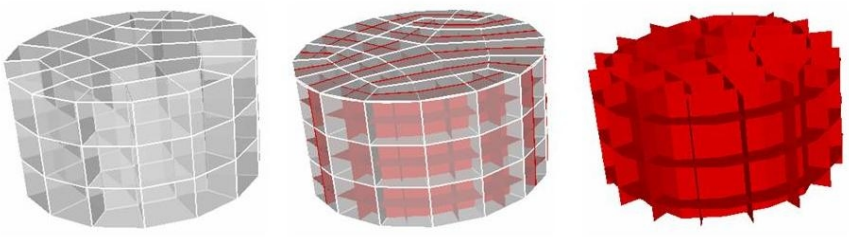

Fig. 2. Example of a cylinder meshed with hexahedra (left). The picture in the middle shows the hex mesh with its dual subdivision in red. The sheets of the mesh are shown on the right.

1. Only three sheets can intersect at a single point. The intersection point of three sheets has been identified as a 'centroid' [39]. A centroid is dual to a single hexahedron in primal mesh.
2. Each sheet in the dual space represents a 'layer' of hexahedral elements in the primal mesh.
3. All sheets either form a closed shell within the mesh space, or have terminating edges around the boundary of the mesh (i.e. there is no sheet that terminates in the interior of the hexahedral mesh.)
4. Sheets cannot be tangent to other sheets.
5. The 'size' of the primal mesh and the number of elements local to an area of the surface is a function of the density of sheets and the triple intersections of sheets relative to that locale within the mesh.

Utilizing these observations along with theorems of topology, Thurston theorized [61] that for a given solid, any quadrilateral mesh composed of an even number of quadrilaterals should admit a compatible hexahedral mesh within the solid. This theory was later proved by Mitchell in [34], and shown to have linear complexity by Eppstein in [15]. These proofs, however, have not resulted in any practical algorithms for generating hexahedral meshes in an arbitrary solid suitable for analytic use.

While it is true that, topologically speaking, any solid on whose boundary contains a quadrilateral mesh with even parity will admit a compatible hexahedral mesh, there must also exist some geometric and quality requirements that must be satisfied to enable a hexahedral mesh to be usable for analytic methods, such as finite element analysis.

3 METHODS

In this section, we explore and derive some of the criteria which must be satisfied (i.e. constraints) for generating a hexahedral mesh in an arbitrary geometry. These constraints are identified using the dual structures of a hexahedral mesh which were outlined earlier. Specifically, we will be looking at

the topology and geometry of the hexahedral sheets both interior to a solid, as well as at the boundary of a geometry. By understanding these constraints, we can propose possible extensions to existing algorithms, along with new approaches, to enhance the class of geometries for which existing hexahedral meshing algorithms are applicable. In the last part of this section we will also highlight some algorithms that may be utilized to satisfy some additional hexahedral constraints within an existing mesh.

3.1 Hexahedral Mesh Constraints

Topological Constraints

In delineating the topological requirements of a hexahedral mesh, we will utilize the constraints given by Mitchell in his proof of hexahedral mesh existence [34]. While his proof started by considering a solid homeomorphic to a ball with an even-parity quadrilateral mesh on the boundary, it should be recognized that the constraints given will also apply to any arrangement of sheets within a solid where no boundary mesh on that solid has been specified. That is, if we start with an arrangement of sheets and place these sheets interior to a solid, the topological constraints enumerated below will still hold with only minor concessions to incorporate the boundary of the solid that does not contain a quadrilateral mesh. The requirements for sheets near the geometric boundary will be discussed in the next section.

Before outlining the topological constraints, we will define additional needed terminology. First, let us clarify our definition of a sheet to be the following: For a given problem domain in $\Re^3$ (where the boundary of the space is defined by the boundary of the solid(s) to be meshed), a sheet is a manifold surface in $\Re^3$. A sheet manifold may be either orientable or non-orientable (Schwartz and Ziegler recently demonstrated an embedded mesh containing a non-orientable sheet [49]). Because the sheet is a manifold in the space, the sheet must either be closed within the boundary of the mesh space or the boundary of the sheet must coincide with the boundary of the mesh space. A collection of sheets will intersect to form various cell complexes. The sheets are restricted in their intersections with other sheets such that the resulting cell complexes are dual to a hexahedral mesh. The following collection of sub-entities will be found in the resulting cell complexes, defined as follows (see also Figure 3):

- A centroid (i.e. a 0D element) occurs at the intersection (local) of three sheets. (It is possible that a single sheet may self-intersect, such that a single sheet is effectively two (or even all three) of the local sheets needed to form a centroid.)
- A chord (i.e. a 1D element) is produced along the intersection (local) of two sheets.

- A 2-cell (i.e. a 2D element) is a polytope on a sheet resulting from an intersection with one, or more, other sheets, where the polytope boundary are the chords produced by the sheet intersection.
- A 3-cell (i.e. a 3D element) is a volumetric polytope resulting from the division of the original space by one, or more, sheets, where the boundary of the 3-cell are the 2-cells enclosing the sub-space.

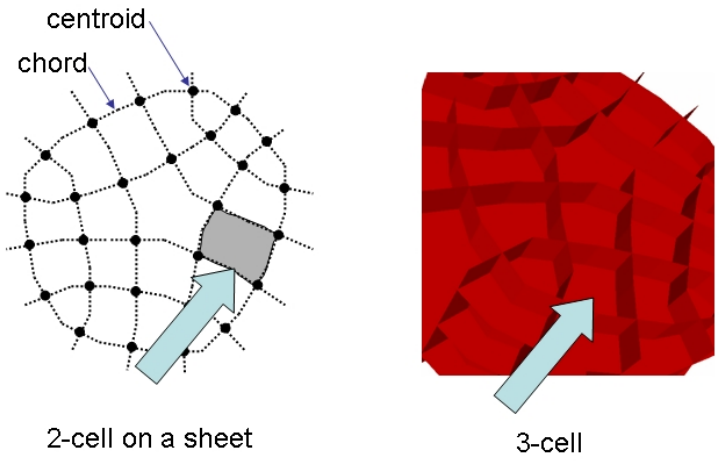

Fig. 3. Entities in the Dual

Utilizing the above definitions, Mitchell outlined [34] the topological requirements for a given arrangement of sheets to produce a hex mesh as follows:

1. Each internal 2-cell is contained in exactly two distinct 3-cells.
2. Each face contains at least one lower dimensional face (excepting centroids).
3. Each chord segment must contain two distinct centroids.
4. Every internal cell contains at most one surface cell of one lower dimension.
5. Each internal chord segment must be contained in exactly four distinct 2-cells.
6. Each centroid is contained in six chord segments. Note, also, that each chord segment at a centroid is paired with another chord segment belonging to the same chord.
7. Two 3-cells have, at most, one 2-cell in common.

When any of the conditions above are violated, the result will be either a degeneracy or void regions in the resulting hexahedral mesh. Proofs for each individual requirement can be found in [34].

For purposes of aiding the reader's intuition, these constraints can, in most cases, be viewed as:

1. Only three sheets can intersect at any given centroid.
2. Sheets cannot be tangent with another sheet.
3. Sheets must span the space, or form a closed surface within the space (i.e. the boundary of the sheet must either coincide with the boundary of the space being meshed, or form a closed manifold within the space.)
4. When traversing the centroids along a single chord, consecutive instances of a single centroid are not permitted (i.e. for the six quadrilaterals of a hexahedron, any two of the six quadrilaterals may not be identical.)

Boundary Constraints

The topological constraints outlined above do not address some of the complexities near the geometric boundary of the space or solid. In this section, we make several observations regarding the geometric boundary of solids and formulate the additional requirements necessary to ensure compatibility of the interior sheets with the geometric boundary of a space or solid geometry.

For clarity, we need to define what is meant by a solid geometry. A solid geometry consists of five main entity types, namely vertices, curves, surfaces, solids (or volumes), and collections of volumes (these may be referred to as an 'assembly'). These entities are often arranged in a hierarchical structure, as shown in Table 1. This hierarchical structure is often referred to as the topology of the solid geometry. (Please note the distinction between the mesh topology and the geometric topology of the solid.) The solid geometry defines the space which will be discretized into a mesh.

While there may be special cases to the hierarchy shown in the table (i.e. curves without endpoints, surfaces without curves, surfaces with zero area, etc.), most of these special cases can be ignored, or remedied by introducing the necessary entities to match the definitions shown in the table and minimally affecting the solid geometry representation.

Geometric Entity	Bounding Entities
Vertex	None
Curve	Two Vertices
Surface	One or more Curves
Volume	One or more Surfaces

Table 1. Hierarchical arrangement of geometric entities.

Understanding the interaction between these entity types is important to understanding how the boundary constraints on hexahedral meshes help to capture the geometric entities. For instance, in order to mesh a surface, the mesh topology of the surface must somehow incorporate the curves and

vertices 'owned' by that surface. For many solid geometries, it may be easy to capture the geometric shape of the object, but very difficult to capture the geometric topology of the object.

One other important distinction is the difference between mesh entities and geometric entities. We will utilize the following terminology, shown in Table 2, in order to distinguish between mesh entities and geometric entities. From this table, we can also note the hierarchical relationship inherent in the mesh entities, which is similar to the relationship between geometric entities noted earlier.

Dimensionality	Geometric Entity	Mesh Entity
0D	Vertex	Node
1D	Curve	Edge
2D	Surface	Quadrilateral (or Triangle, etc.)
3D	Volume	Hexahedron (or Tetrahedron, etc.)

Table 2. Relationships between geometric entities and mesh entities.

It is also possible to construct a table (Table 3) indicating entity correspondence between the dual and primal spaces for hexahedral meshes.

Primal Entity	Dimension	Dual Entity	Dimension
Hex	3	Centroid	0
Quad	2	Chord	1
Edge	1	2-Cell	2
Node	0	3-Cell	3

Table 3. Conversions between dual and primal entities.

Geometric Surface Constraints

A couple of key observations can be made from Table 3. First, note that a chord is dual to a quadrilateral. Since chords were drawn between centers of hexahedra, the chord between two centroids was dual to the quad shared by these two hexahedra. In some sense, the chord can be viewed as an approximation to the normal of the quadrilateral between two hexes.

Observation 1:

The boundary of any hexahedral mesh is a quadrilateral mesh.

To improve the quality of the mesh at the boundary of our geometry, it is desireable to align the chords with the local surface normals. Doing this promotes higher quality hexahedral elements on the boundary of the mesh.

Boundary Constraint 1:

For each surface of a hexahedrally meshed solid, there exists a set of sheet patches, which are geometrically similar (i.e. similar in shape) to the surface, but offset a distance that is a function of the size of the mesh local to that boundary (i.e. the local chord length). The minimal number of sheets to which the collection of sheet patches might belong is a single sheet.

Justification:

Given a hexahedral mesh and utilizing Observation 1, select a quadrilateral on the boundary. This quadrilateral is dual to a chord that is found at the intersection of two sheets whose boundary are the two chords intersecting the quadrilateral on the boundary. Since the boundary quadrilateral is contained in only one hexahedral element of the mesh, there must be a third sheet that intersects the other two creating the centroid that is dual to the hex. On this third sheet, there is an area within the sheet that can be said to correspond directly with the quadrilateral on the boundary. This area of the sheet is also geometrically similar in shape to the quadrilateral, although offset a distance based on the size of the hexahedron to which the sheet and quadrilateral correspond. A collection of contiguous areas on a single sheet corresponding to a set of quadrilaterals on the boundary will be defined as a 'sheet patch'. The collection of 'sheet patches' corresponding to all quadrilaterals on the surface boundary is geometrically similar to the boundary surface. (see Figure 4). Also notice that the minimal number of sheets to which this collection of patches might belong is a single sheet.

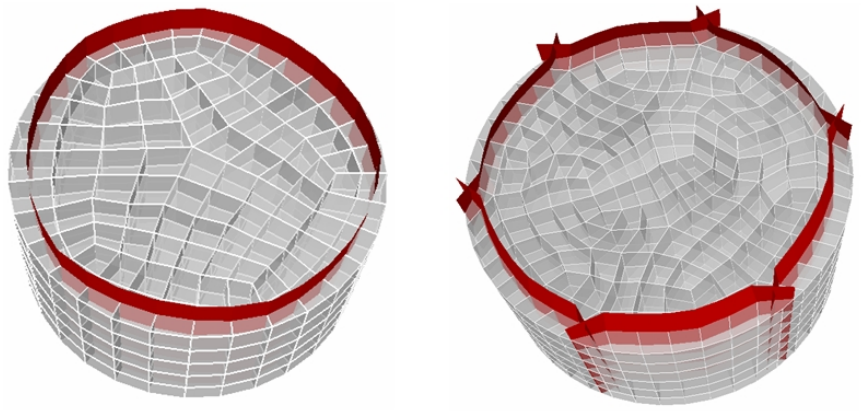

Fig. 4. Image showing how a sheet captures the geometric boundary. The picture on the right shows a single sheet capturing the cylindrical surface, while the picture on the left (of a different mesh) shows the same surface being captured with multiple sheets.

This poses an immediate question: Is it better to capture a boundary surface with a single sheet, or with multiple sheet patches? For the majority of cases, the answer to this question is that single sheets for capturing a surface are better than multiple sheet patches. The reasoning for this is due to the irregularities introduced whenever a sheet is diverted away from a surface. When partial sheets are utilized to capture a surface, the area of the sheet where the transition occurs results in increased nodal valence in the mesh, skewing of the elements (caused by the sheet curvature), and in some cases the formation of adjacent hexahedron that contain faces sharing two or more edges (also known as a 'doublet' and will be described in more detail later in the paper). Figure 5 shows several examples of the transition elements resulting from partial sheets capturing boundary surfaces. Therefore, in most cases, the use of a single sheet to capture boundary surfaces is preferred.

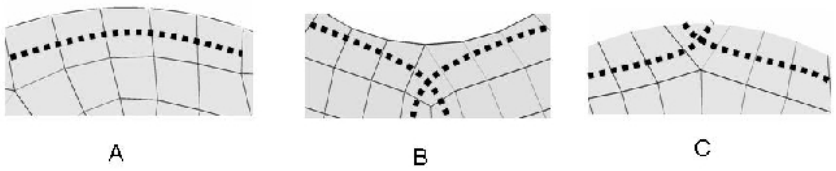

Fig. 5. When partial sheets capture boundaries, the quality and regularity of the mesh is affected. Image A shows elements from a single sheet capturing the upper boundary of the solid. Image B and C use patches from two sheets to capture the upper boundary of the solid. In Image B and C, note how the regularity of the mesh is affected, and the resulting skew in the transition element due to the sheet curvature away from the boundary. In Image C, a near doublet element is formed due to the low curvature of the boundary being captured.

Geometric Curve Constraints

Boundary Constraint 2:

For each curve on a hexahedrally meshed solid, there exists a set of sheet patch pairs such that the lines of intersection between each of the patch pairs is a piecewise approximation of the curve, only offset a distance, which is a function of the mesh sizes local to the curve.

Justification:

From our description of curves in solid geometry, a curve in the solid is the boundary between two surfaces on the solid. From Boundary Constraint 1, there exists a collection of sheet patches local to the curve that are geometrically similar to each of the surfaces. The lines of intersection of two sets sheet patches are a set of chord segments that are geometrically similar to the curve

at the boundary between the two surfaces, and that are offset a distance which is a function of the mesh size in the boundary curve's locale (see Figure 6):

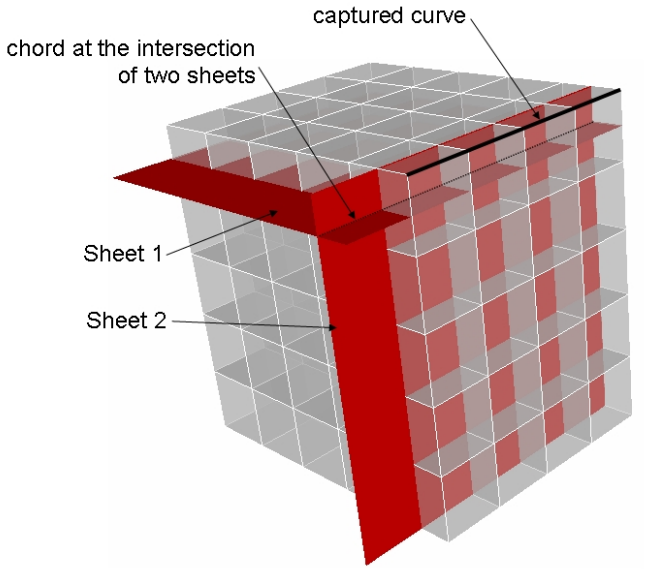

Fig. 6. Capturing a curve utilizing an offset chord (from the intersection of two sheets).

Geometric Vertex Constraints

Boundary Constraint 3:

There exists at least one triple-sheet pairing that corresponds to each vertex on the boundary. This triple-sheet pair is equivalent to a centroid, and is offset a distance related to the mesh size local to the vertex.

Justification:

Any vertex on the boundary of the solid must contain one mesh node in order for the mesh to be compatible with the solid. Since a node is dual to a 3-cell, and the simplest 3-cell is formed at the intersection of three sheets with the boundary of the solid. Therefore, there exists at least one-triple sheet pairing that corresponds to each vertex on the boundary.

Some additional insights may help better understand geometric vertex constraints. Because a node is at the corner of each hexahedral element, we can also draw relationships for each vertex to a set of centroids. Because every node is contained in at least one hexahedral element, there must exist at least

one centroid that is offset a distance related to the mesh size local to the vertex (the intersection of the triple-sheet pair is the centroid corresponding to the vertex in the simplest case). However, there is a complicating difference with vertices, in that it is not necessary for a vertex to correspond to a single centroid. Rather, a vertex may correspond to many centroids within a single solid.

We can re-write this observation in terms of sheets, such that, for each centroid, there exist a local, triple-set of sheets whose intersection form a centroid that corresponds to a vertex. However, because a vertex on the boundary can correspond to one or more centroids, there will be cases when this vertex will correspond to more than one triple-sheet pairing. A few examples are shown in Figure 7.

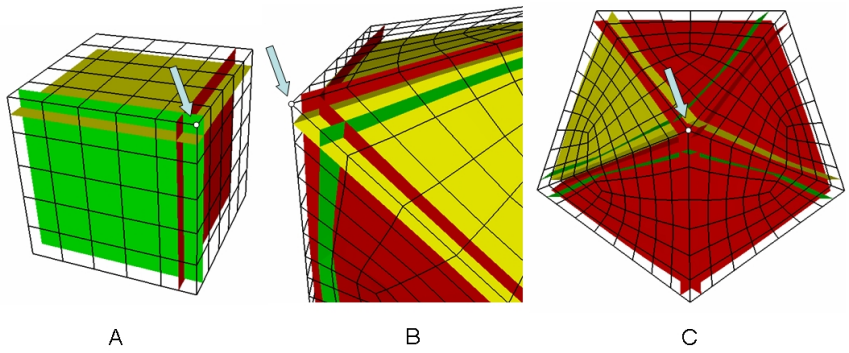

Fig. 7. At least three sheets are necessary to capture a geometric vertex in a given mesh topology (A). However, more than one triple-pair of sheets is not exclusive for each vertex. For geometric vertices whose valence is higher than three, more than one triple pair is necessary to capture all of the geometric features related to the vertex. A four-sided pyramid (B) requires four sheets (creating two distinct centroids, or two triple-pairs) and a five-sided pyramid (C) requires five sheets (creating three distinct centroids, or three triple-pairs) to succinctly capture the geometric features associated with the vertex. In (B), there are two red sheets, one green, and one yellow shown. In (C), there are two red, two green and one yellow sheet shown.

In this case, utilizing the sheets in meeting this constraint is helpful instead of the centroids because there are usually geometric curves emanating from each of the vertices. The sheets utilized to capture the geometric surfaces and curves will also be the same sheets which capture the vertices. The convergence of many sheets around some vertices must be handled with care to maintain the topological constraints local to a vertex (i.e. only three sheets can intersect at a single point, etc.).

Geometric, or Quality, Constraints

Specifying constraints in relation to the quality of the mesh is often difficult because a complete understanding of how mesh quality affects analysis error is largely unknown, specifically with hexahedral meshes. While there has been some recent work in this area [53, 22], the only consistent constraint placed on hexahedral mesh generation is that the (scaled) Jacobian of each mesh element must be positive (i.e. non-inverted). In this section, we will demonstrate how the Jacobian for a hexahedral mesh is defined, and we will consider how the geometric properties of a sheet affect the overall quality of the resulting hex mesh, while the ultimate goal will be to determine the necessary conditions on a sheet to have a resulting non-inverted mesh. We will begin by first discussing the sheets relating to the ideal mesh as developed for finite elements, and then by discussing how geometric and topological modifications to the ideal sheets change the underlying quality of the resulting hexahedral meshes.

Quality Considerations

In terms of hexahedra, the ideal element for which finite element basis functions are formulated is a cube of six quadrilaterals of equal area, with edges of equal length, and which are mutually orthogonal to each other. Such cubes are easily subdivided by dividing each edge by two and each quadrilateral into four smaller, but mutually equivalent quadrilaterals. The arrangement of the sheets in dual space for such an ideal mesh is simply a collection of Cartesian planes which subdivide the mesh as described earlier in the dual construction of a mesh (see Figure 8).

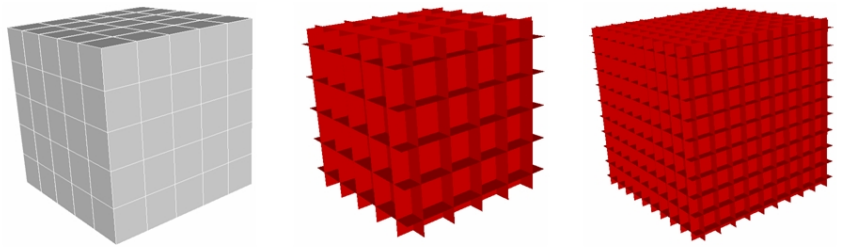

Fig. 8. The ideal mesh (left) with the induced sheet arrangements at two different mesh sizes (the sheet arrangement on the right is twice as dense as the sheet arrangement in the middle).

Utilizing the ideal mesh as our goal, the following constraints can be added:

1. **Maximize orthogonality in the sheet topology -** From the topological constraints earlier, we know that at most three sheets can intersect at a

single point. From the ideal mesh, it is apparent that a perfectly orthogonal intersection of the three sheets is desired. Deviations from orthogonal intersections of the sheets will produce 'skewing' of the hexahedral elements in the mesh. (see Figure 9).

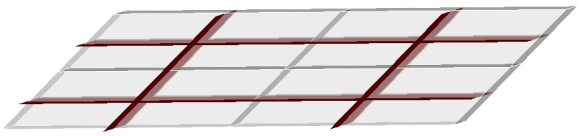

Fig. 9. As sheet intersections deviate from orthogonality, the skew of the mesh increases.

2. **Minimize sheet curvature -** The ideal isotropic mesh contains perfectly planar sheets. Therefore, it is desireable for the local curvature of a sheet to be as minimal as possible. Curvature of the sheets causes 'keystoning' of the elements, where the length of one edge is substantially different than it's opposite edge. This phenomenon can be readily seen in Figure 10, as we increase the curvature of a single sheet of elements.
3. **Maximize the topologic regularity of the sheets -** The regular topological arrangement of the sheets, as shown in the ideal mesh, is also desirable (and is essentially required in finite difference calculations). In finite element methods this requirement has some flexibility, but maintenance of this regular topology, where possible, has some additional benefits from several algorithmic standpoints including element numbering, matrix formulation, compression, etc. Additionally, non-regular arrangements of sheets have a tendency to interfere with the optimization of constraints 1 and 2 above.
4. **Sheet density controls element sizing -** Element sizing information is determined directionally as a function of the local density of the sheets in an area. To increase the element size in one direction, decrease the density of locally parallel arrangement of sheets. Except in a few cases (for example, boundary layers in fluid flow), a dramatic transition in the density of sheets is undesirable (see Figure 11).

Mesh Untangling

Before proceeding further, it will be helpful to introduce the idea of mesh untangling as developed by Knupp, et al. [21]. The Jacobian matrix is calculated for each node with respect to a hexahedral element. For each node in a hex, there are exactly three 'neighbor' nodes connected via an edge in the hexahedra. For a single node of a hexahedra, the Jacobian matrix is defined as:

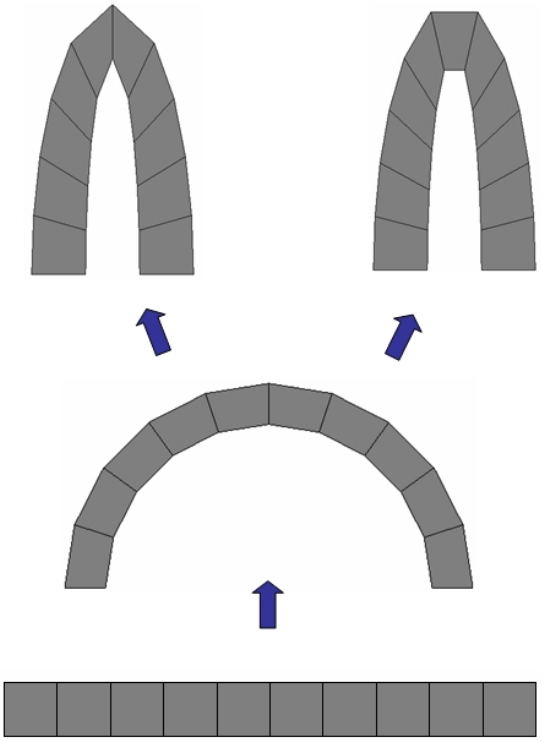

Fig. 10. Increasing the curvature of a single sheet results in 'keystoning' of elements where one edge shrinks in size while the other grows as the curvature of the sheet increases.

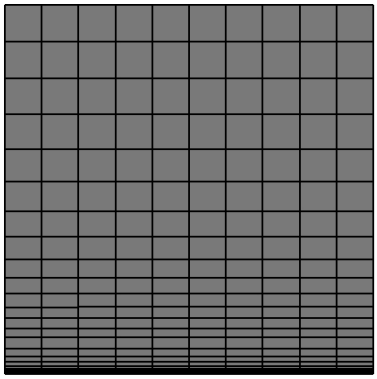

Fig. 11. Note that as the sheet density increases in one direction only, the element aspect ratio increases as can be seen in this figure.

$$A_0 = \begin{vmatrix} x_1 - x_0 & x_2 - x_0 & x_3 - x_0 \\ y_1 - y_0 & y_2 - y_0 & y_3 - y_0 \\ z_1 - z_0 & z_2 - z_0 & z_3 - z_0 \end{vmatrix}$$

For a single hex, there will be eight such matrices (for additional discussion on elements with multiple Jacobian matrices see [22]. The minimal determinant of these eight matrices is known as the 'Jacobian' of the hexahedral element. By allowing nodal movement for each of these elements, an optimization problem can be formulated to maximize the following objective function (other similar functions have also been utilized):

$$f(A) = 0.5 * \sum(|\alpha_m| - \alpha_m)$$

where α_m is the determinant a single Jacobian for a mesh with m elements.

If the mesh is untangled, then the summation reaches a maximum value of zero. For a hexahedral mesh, this can be a difficult optimization. It is common to use local optimization algorithms, such as conjugate gradient methods, to obtain a solution to the untangling optimization problem. However, because the untangling problem is non-convex, it is possible to reach a local maxima without obtaining an optimal solution. This is an ongoing and challenging research area [21, 62, 23, 17]. It is undetermined whether a mesh that satisfies the topological requirements cited earlier can always be untangled, although it is believed that there are meshes that do not have an untangled solution.

Additional Quality Observations

The quality of a hexahedral mesh, as determined from its dual subdivision, is largely unexplored. From the quality considerations enumerated earlier, it is known that there exists a correlation between the conformation of the sheet and the ultimate quality of the hexahedral mesh. Specifically, high curvature of the sheet is a necessary condition to creating an inverted hexahedral element, however, as shown earlier, high curvature of the sheet is not a sufficient condition to guarantee that elements will become inverted. We list some case studies that may be of value in determining the additional sufficient conditions resulting in poor quality hexahedral meshes.

- *Low sheet curvature is important for high quality* - Figure 12 shows a sheet from a volume that was meshed via the whisker weaving algorithm [16]. At the base of the trough in this sheet is a collection of inverted elements that are currently untangle-able [21, 62]. Mesh smoothing can only improve the sheet conformation in a limited fashion because of the fixed mesh topology the smoothing algorithms must maintain while adjusting the conformation of one sheet with the surrounding sheets. Some efforts involving mesh topology reconfiguration (for instance, mesh-flipping [3, 4, 58]) may aid our ability to untangle these meshes by relaxing the fixed topology of the mesh enabling sheet conformations with lower curvature to be obtained.

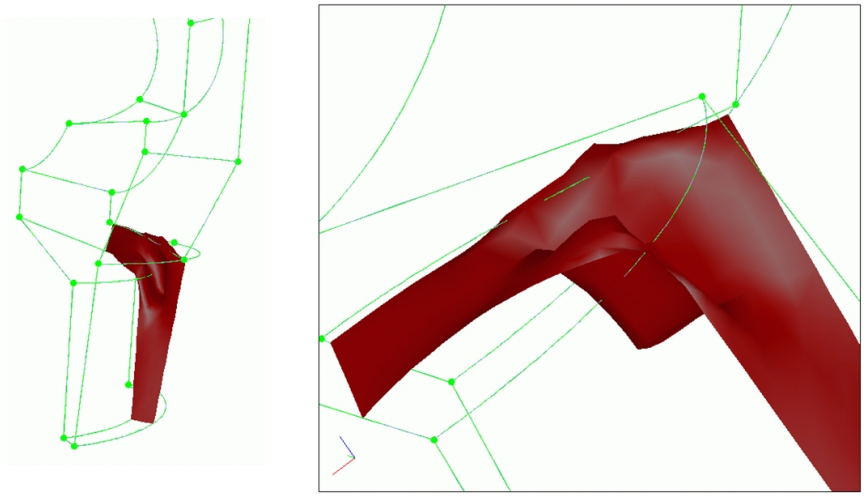

Fig. 12. Two views of a sheet with a high curvature trough in a solid. The high curvature at the trough of the sheet shown corresponds to hex elements that cannot be untangled in the resulting mesh.

- *The surface mesh is important for high quality* - The boundary of a non-closed sheet in a hexahedral mesh is defined by a dual cycle of chords in the quadrilateral mesh on the boundary. Some of the dual cycles defined by quadrilateral meshes imply complex geometric definitions for any resulting sheet corresponding to the defined dual cycle boundary. One such boundary is shown in Figure 13, where the dual cycle wraps around itself several times in one area of the solid. Obvious high curvature in the sheets and coarse interactions between adjacent sheets, results in a mesh for which an untangled solution is not currently known.
- *The mesh approximates the sheet* - Recent work done by Suzuki, et al. [56] has explicitly defined the sheets interior to a solid. By generating an interior surface for the dual cycle of Schneider's pyramid [48] and subsequently satisfying all necessary topological constraints for a hex mesh, it was determined that a reasonable quality mesh would result. However, the resulting mesh had elements that were untangle-able. Because the sheet was defined geometrically prior to creating the mesh, we can compare the sheet defined by the hex elements with the sheet created by the authors. The differences between the two are significant and the sheet defined by the mesh is a very rough approximation of the desired interior surface (see Figure 14. Increasing the number of elements can dramatically improve some areas of quality, but all of resulting meshes do not have a known untangled solution.

These case studies indicate that there is still a great deal of information that is not currently understood with regards to generation of quality (or, at

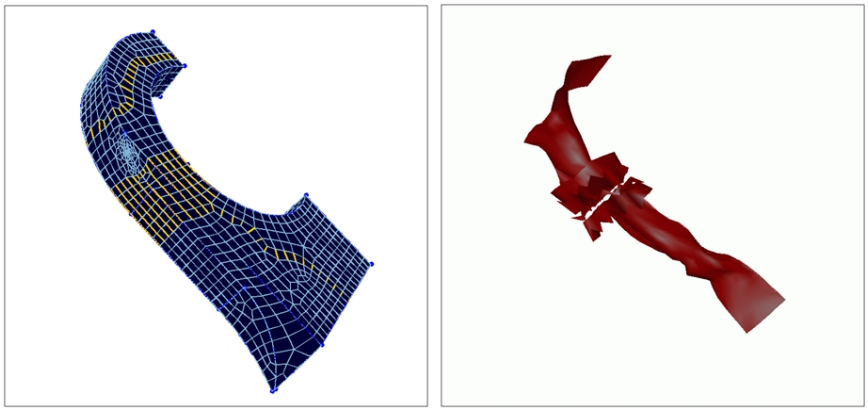

Fig. 13. The surface mesh (shown on the left) has a dual cycle that spirals up to the top of the image. The resulting hex mesh, generated with WhiskerWeaving, cannot be untangled. The sheet generated by WhiskerWeaving is shown on the right.

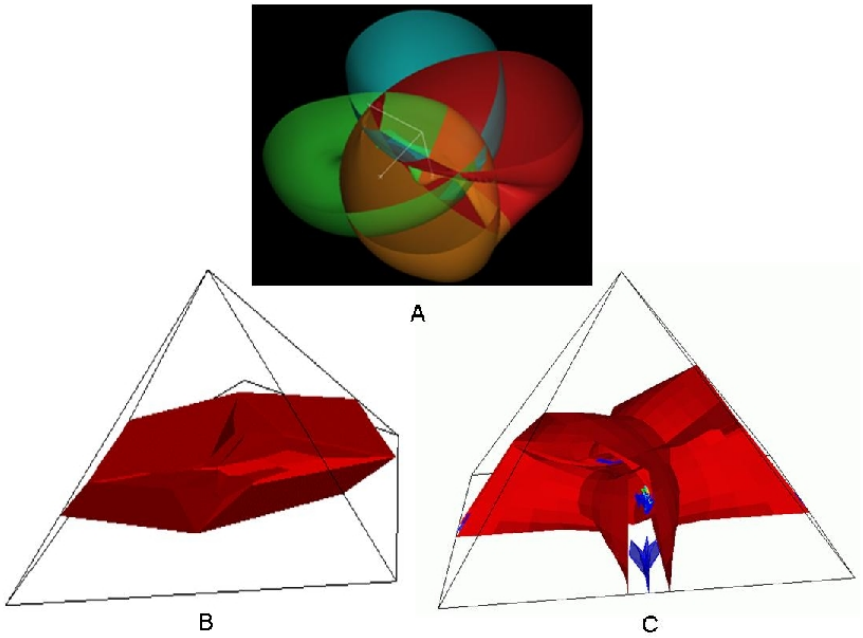

Fig. 14. The boundary and interior surface, shown in (A), correspond to the boundary and an interior surface for Schneider's pyramid [48]. The resulting hex mesh has a sheet (B) which approximates the surface in (A), but the facets tend to flatten the intended sheet conformation. By increasing the resolution (via dicing [32]) the approximation of the surface is better and has fewer regions of negative Jacobian elements (some negative Jacobian elements are shown in the image).

least, non-inverted) hexahedral meshes. We know that high sheet curvature is necessary for producing inverted elements, but it is not sufficient. There are obviously some additional sufficient conditions, respect to how multiple sheets with high curvature interact with each other in order to produce sets of inverted elements.

3.2 Constraint-satisfying Methods

Over the years, several methods have been developed that make it possible to improve the flexibility of existing hexahedral meshing algorithms and aid in capturing constraints overlooked by the paradigm employed by a specific algorithm. In this section, we highlight some of these methods and show how they help to satisfy some of the fundamental constraints for hexahedral meshing of solid models. In particular, we discuss methods for inserting and extracting sheets in existing mesh topologies. The methods we highlight are pillowing [37], dicing [32, 31], mesh-cutting [9], grafting [19], and sheet extraction [8].

Inserting Sheets

Pillowing

During the development of the whisker weaving algorithm [57, 16], Mitchell et al. consistently encountered meshes where two neighboring hexes shared two faces (or more basically, where two adjacent faces shared two edges). Called a 'doublet' (see Figure 15), this situation is undesirable because of the impossibility of moving the nodal locations in these elements such that both elements have positive Jacobian determinant values. A simple, but powerful, technique called 'pillowing' [37] was developed to locate and place a mesh refinement that effectively removed the doublets from the mesh.

In terms of the dual of the mesh, pillowing is essentially a sheet insertion operation, where a new sheet is inserted that effectively splits the doublet hexes into multiple hexes, and eliminating the problematic mesh topology.

The pillowing method turns out to be powerful, not for it's ability to remove doublets, rather it provides a fairly straight-forward approach to insert sheets into existing meshes. The sheets can be inserted utilizing the primal elements of an existing mesh, and without explicitly creating a geometric definition for the sheet and calculating the intersections with the other local sheets in the space.

The basic pillowing algorithm is as follows:

1. *Define a shrink set* - For our purposes, this step involves dividing the existing mesh into two sets of elements: one set for each of the half-spaces defined by the sheet to be inserted. One of these two sets of hexahedral elements comprises the shrink set. The choice of which one should be the shrink set is arbitrary, although the best algorithmic choice will be the set with the fewest number of elements.

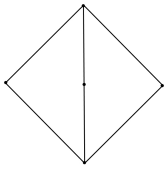

Fig. 15. A quadrilateral doublet, where two adjacent quadrilaterals share two edges. Similar types of doublets occur in 3D with adjacent hexes sharing two or more quadrilaterals. The scaled Jacobian for both elements, as shown, is zero, and while node movement strategies can improve the Jacobian value for one of the two quadrilaterals, simultaneous improvement of the Jacobian value for both quadrilaterals is not possible.

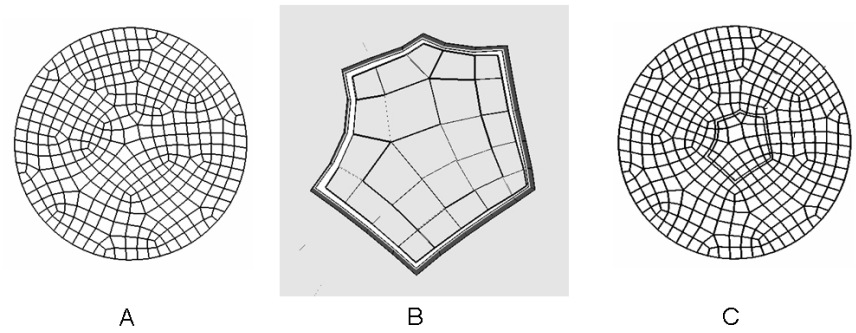

Fig. 16. A basic pillowing operation starts with an initial mesh (A) from which a subset of elements is defined to create a shrink set. The shrink set is separated from the original mesh and 'shrunk' (B), and a new layer of elements (i.e. a dual sheet) is inserted (C) to fill the void left by the shrinking process.

2. *Shrink the shrink set* - This step essentially creates a gap region between the two previous element sets (see Figure 16. The difficulty in this step involves splitting the shared nodes, edges, and quads in the existing mesh, while maintaining the appropriate correspondence of the mesh entities with the geometric topology.
3. *Connect with a layer of elements* - This step results in a fully-conformal mesh with the new sheet inserted between the original two element sets. To complete this step, an edge is inserted between each node that was separated during the 'shrinking' operation. Utilizing the quadrilaterals on the boundary between the two sets of hexes, along with these new edges, it is fairly straight-forward to determine the connectivity of all of the hexes in this new layer.

It is often desirable to perform a smoothing operation on the resulting mesh after the new sheet has been inserted to obtain better nodal placement and higher quality elements. The speed of the pillowing algorithm is largely

dependent on the time needed to find the shrink set. The number of new hexahedra created will be equal to the number of quadrilaterals on the boundary of the shrink set.

A pillowing operation is nothing more than a sheet insertion operation for an existing mesh. It is therefore a useful multi-purpose, foundational tool for operations on hexahedral meshes, including doublet removal, refinement, grafting, and mesh-cutting (grafting and mesh-cutting will be discussed in upcoming sections.)

Dicing

The dicing algorithm [32, 31] was created to very efficiently generate very large, refined meshes from existing coarse meshes. The dicing method is an efficient tool for duplicating sheets multiple times within an existing mesh. The generation of these very large, refined meshes is accomplished by copying each of the existing sheets and placing them in a parallel configuration to the sheet being copied. The basic method for dicing is as follows:

1. *Define the sheet to be diced* - An edge in a hexahedral mesh can only correspond to a single sheet in the dual. Utilizing one edge, the opposite edges of the hexahedron can be deduced as belonging to the same sheet via the definition of the dual of the hexahedral mesh. It is then possible to iterate until all of the edges associated with a single sheet in the dual are found.
2. *Dice the edges* - With the list of edges found in the previous step, dicing then splits (dices) all of these edges the specified number of times. If for instance, we wish to copy the sheet one time, then each of the edges is split once resulting in two new edges.
3. *Form the new sheets* - With each of the edges associated with the hexahedral sheet split, we can again utilize the idea that an edge can be associated with a single sheet in the mesh and form a new layer of hexahedra for each split in the original set of edges, where the hexahedral connectivity is similar to the original hexahedral layer before the edges were split.

Utilizing the dicing method, the number of elements increases as the cube of the dicing value. For instance, if an existing mesh is diced four times (i.e. each of the sheets in the existing mesh is copied four times), the resulting mesh would have 64X as many elements as the original mesh. Because all search operations can be performed directly, the dicing algorithm can produce large meshes at very efficient speeds (see Figure 17).

Geometric Capture with Sheets

Mesh Cutting

The mesh-cutting [9] method is an effective approach for capturing geometric surfaces within an existing mesh topology. The mesh-cutting method utilizes

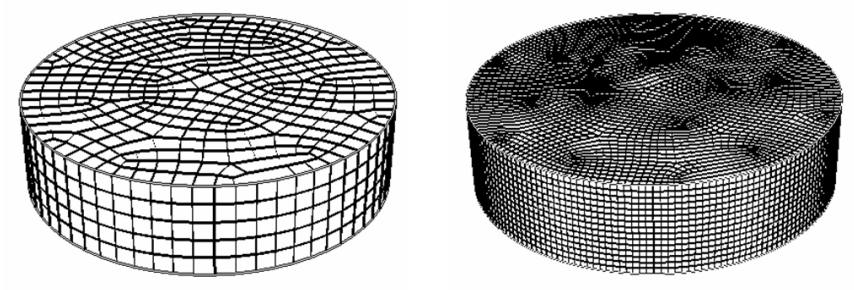

Fig. 17. The original mesh (left) contains 1805 hex elements before dicing. Each sheet in the original mesh is copied three times resulting in a mesh that is 3^3 larger, with 48,735 hex elements.

the pillowing and dicing methods mentioned previously to insert two sheets which are geometrically similar to the surface to be captured. By utilizing two sheets, the result is a layer of quadrilaterals, shared by the hexes in the two sheets, which can be viewed as a set of facets geometrically approximating the surface. The mesh-cutting method entails the following steps:

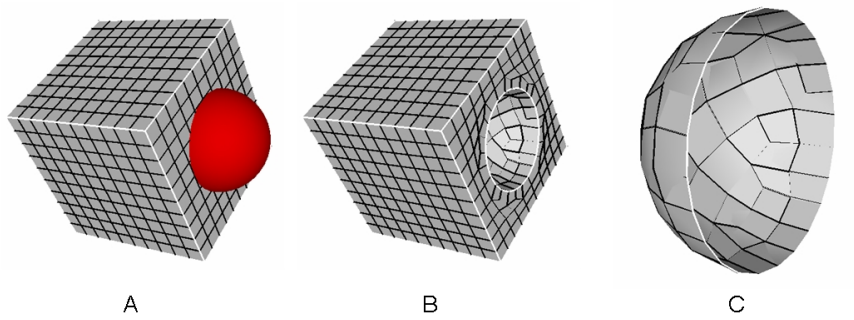

Fig. 18. Mesh cutting utilizes an existing mesh and inserts new sheets to capture a geometric surface(the existing mesh is shown in (A) where the red, spherical surface is the surface to be captured.) The resulting mesh after mesh cutting is shown in (B), with a close-up of the quadrilaterals on the captured surface being shown in (C).

1. *Define the pillowing shrink set* - Utilizing the surface that is to be captured in the mesh, we divide the existing mesh into two sets of elements. One of these element sets will be the shrink set, and a sheet (pillow) is inserted between the two sets of elements.
2. *Dice the new sheet* - We split the newly inserted sheet into two sheets utilizing an approach similar to dicing.

3. *Move the shared quadrilaterals to the surface* - With two new sheets defined in the mesh topology, we can find all of the quadrilaterals that are shared by the hexes between the two sheets. These quadrilaterals become the mesh on the surface we are attempting to capture (see Figure 18).

A caveat with this method is that the existing mesh topology must be fine enough to capture the detail of the surface to be inserted. Because the resulting quadrilaterals only approximate the inserted surface, if the resulting quadrilateral mesh is too coarse, the surface may not be approximated adequately enough to be resolved.

One other item to remember with this method is that since a geometric surface is being utilized to define a sheet within the mesh space it may be necessary to have the ability to extend the geometric surface in some fashion such that it meets the requirements on a sheet that it divide the mesh space. If the geometric surface is trimmed, for instance, the trimmed surface may not adequately divide the space being meshed making it necessary to provide a continuation to the surface definition to the boundary of the mesh space.

Grafting

The term 'grafting' is derived from the process of grafting a branch from one tree into the stem, or trunk, of another tree. In meshing, the grafting method was initially to be utilized for allowing a branch mesh to be inserted into the linking surface of a previously hexahedrally swept volume [19]. The grafting method would then offer a reasonably generalized approach to multi-axis sweeping.

Grafting is a method that essentially captures geometric curves on previously meshed surfaces. Because the curve(s) already reside on a surface and the existing mesh already captures that surface, it is possible to introduce a second sheet to satisfy the curve constraint listed earlier. The introduction of the new sheet will produce the necessary mesh topology to enable the mesh to be compatible with the boundary curve(s). The method for creating a graft (i.e. capturing the geometric curve) can be outlined as follows (see also Figure 19):

1. *Create a pillowing shrink set* - In the case of grafting, the shrink set is typically defined as the set of hexes which have one quadrilateral owned by the surface and which are interior to the closed set of curves (with respect to the surface).
2. *Insert the pillow (sheet)* - By inserting the second sheet, we have essentially satisfied the hexahedral constraint for capturing a geometric curve. That is, we now have two sheets which generate a chord in the mesh that is offset from the set of curves which were the input to the grafting algorithm.

At this point, there is often some database adjustments also necessary to ensure that the new mesh entities are associated with the correct geometric

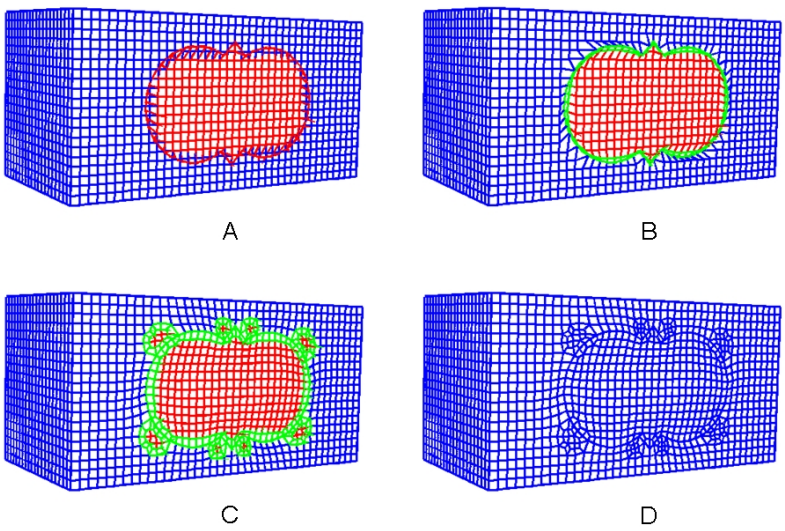

Fig. 19. In grafting, a shrink set on a existing meshed volume is defined (A) and a new sheet is inserted via a pillowing operation (B). Once the new sheet has been inserted, the nodes are positioned along the curve to be captured via a smoothing operation (C). Additional pillows can also be inserted to remove any doublet hexes that may have been created (C). The resulting mesh topology captures the geometric curve (D).

entities, but the curve is essentially captured when the second sheet is inserted in conjunction with the initial set of sheets that captured the geometric surface. A similar method can be used to capture a single curve, rather than a set of curves, but it is still necessary that the sheet that is inserted to capture the curve satisfy the definition of a sheet. That is, the sheet must completely divide the mesh space into two regions.

There is one caveat with this method: Because there is no explicit steps taken to capture the geometric vertices associated with each of the curves being grafted, there is a requirement that the resolution of the trunk mesh be fine enough to be able to capture all of the curve's endpoints by moving existing nodes in the final mesh to the vertex locations. The movement of the nodes to the vertex locations must be done intelligently to avoid destroying the required mesh topology necessary to correctly capture the curve. While other sheets may be added to avoid this problem, the addition of more sheets to capture the vertices may have the negative effect of locally refining the mesh sizes and/or mesh topologies that are not as aesthetically pleasing as may be desired.

Removing Sheets

Sheet Extraction

One of the positive practical benefits about working with the sheets in hexahedral meshing, is that all of the processes are easily reversible. Sheet extraction is the inverse operation to a sheet insertion operation (i.e. pillowing or dicing). A method for extracting a sheet is detailed in [8], where the basic steps can be outlined as follows:

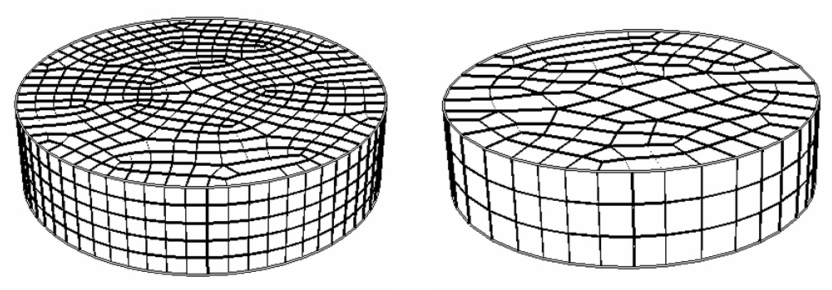

Fig. 20. Original mesh, shown on left, with 1805 hex elements. After removing approximately half of the sheets in the original mesh, the resulting mesh (right) has 342 hex elements.

1. *Define the sheet to be extracted* - Because an edge in a hexahedral mesh can only correspond to a single sheet in the dual, this step can be easily accomplished by specifying a single edge in the mesh. From this single edge, the primal mesh can be iteratively traversed to determine all of the edges which correspond to the sheet to be extracted.
2. *Collapse the edges* - With the list of edges found in the previous step, the nodes of each of the edges can be merged, effectively removing the sheet from the mesh.

There are some special circumstances that must be avoided when extracting sheets in order to avoid either degenerating the mesh or producing a mesh which is no longer conformal with the geometric topology. These situations can be avoided by checking to ensure that one of the edges to be collapsed is not the only edge on a curve, or that the nodes in an edge are not owned by different curves, etc.

4 SUMMARY AND APPLICATION OF CONSTRAINTS

In this section, we summarize the hexahedral mesh generation constraints detailed earlier and highlight how knowledge of these constraints might be

utilized to enable faster, more robust, hexahedral mesh generation with existing algorithms.

4.1 Summary of Constraints

The constraints for hexahedral mesh generation as detailed earlier can be summarized as follows:

- Topologic:
  1. Only three sheets can intersect at any given centroid.
  2. Sheets cannot be tangent with another sheet.
  3. Sheets must span the space, or form a closed surface within the space.
  4. When traversing the centroids along a single chord, consecutive instances of a single centroid are not permitted.
- Boundary:
  1. Each surface of the solid-to-be-meshed must have a set of sheet patches which, collectively, are geometrically similar to the surface but offset interior to the solid. The minimal number of sheets to which the collection of patches might belong is a single sheet.
  2. Each curve of the solid-to-be-meshed must have a set of chord patches (resulting from the intersection of pairs of sheet patches) which, collectively, are a piecewise approximation to the curve offset interior to the solid.
  3. Each vertex of the solid-to-be-meshed must have at least one triple-sheet-pairing (i.e. centroid) which corresponds to the vertex on the solid.
- Geometric or Quality:
  1. Maximize the orthogonality of the sheet intersections.
  2. Minimize curvature of the sheets.
  3. Maximize a regular topologic arrangement of sheet intersections.
  4. Maintain uniform density of sheets throughout the solid (with exceptions where mesh anisotropy is desired).

4.2 Application of Constraints

Owen [41] categorized the classes of hexahedral mesh generation algorithms as follows: Mapped Meshing, Direct Methods, and Indirect Methods. In this section, we briefly examine a few algorithms in each category and suggest how better knowledge or incorporation of the constraints detailed above may extend the class of geometries that the algorithm can successfully mesh.

Mapped Meshing

Mapped meshing algorithms include the octree approaches [47, 73, 50, 74], mapping [13], multi-block [14, 72, 43, 1], submapping [66, 67], and sweeping

algorithms [20, 54, 46, 26, 25, 52, 5]. These algorithms are also categorized as structured, or semi-structured methods because of the regularity of the meshes that they typically produce.

These methods work well for a well-defined set of geometric topologies. For instance, mapping algorithms require geometric topologies conforming to a cuboid (i.e. six square-ish surfaces composed of four curves each form the boundary of the solid), and sweeping algorithms require cylindrical topologies. Algorithms that automatically detect or enforce the pre-defined topologies have significantly improved the time required to generate hexahedral meshes [71, 35, 36, 51, 70].

The most popular, and commonly displayed, types of hexahedral meshes are those utilizing a multiblock method for generating a hexahedral grid [72, 43, 1]. The major advantages of these methods are related to the size of the meshes and relative speed with which large meshes can be constructed once the block decompositions have been defined. An individual skilled in generating multiblock-type grids can generate complex meshes with exceptional quality. Because of the control that is available in the creation and layering of the mesh, this method is the common choice for hexahedral decompositions in fluid dynamic simulations where fine control of the mesh near geometric boundaries is required.

The major disadvantage associated with these algorithms is their seeming inflexibility to incorporate additional geometric topologies to the pre-defined geometric topology written for the algorithm. Therefore, hexahedral meshing of non-conforming geometric topologies results in decomposition of the solid into the pre-defined topologies [7, 28, 59], suppression of incompatible topology [24], pre-defined topology overlays (used extensively in multi-block methods), refinement [64], or massaging the elements to fit the geometry [63, 75].

With these algorithms, satisfaction of the boundary constraints is only guaranteed if the geometric topology matches a pre-defined geometric topology. Potentially, all of these algorithms could be augmented for larger-classes of geometric topologies by enabling the algorithms to capture additional boundary details. Mesh cutting and grafting algorithms have been utilized as a post-meshing method to capture additional curves and surfaces in meshes created by the structured methods cited above [9, 19]. It would also be possible to incorporate some additional paradigms directly into these methods that allow the hexahedral boundary constraints to be satisfied during the course of the initial mesh generation phase.

Direct Methods

The Direct Methods include the following classes of algorithms: grid-based, medial axis, plastering and whisker weaving. The grid-based algorithms suffer from the same boundary constraint problems identified in the structured and semi-structured methods above. The medial-axis algorithms define decompositions of the volume into simpler primitives that can be meshed with

a structured scheme. These methods ultimately have difficulty satisfying the boundary constraints for a hexahedral mesh.

The Plastering [6] and Whisker Weaving [57, 16, 38, 11] algorithms both start with a quadrilaterally meshed boundary, which effectively enables easy satisfaction of the boundary constraints. However, because a boundary mesh also defines the boundaries of the sheets that will be placed on the interior of the volume, the pre-meshed boundaries are also the cause of the poor quality, or subsequent failure of these algorithms. As was previously shown in Figure 13, it is quite common for quadrilateral meshes on the boundary to define sheets with high geometric complexity. Assuming the algorithm is sophisticated enough to solve the topologic requirements to define the sheets, the interactions of these high curvature sheets often prevents reasonable quality hexes from being realized.

Some success has been found by removing the problematic sheets with a sheet extraction algorithm, or placing additional constraints on the quadrilateral mesh placed on the boundary (in particular, Hannemann [38] was able to generate some relatively complex hexahedral meshes by utilizing quadrangular patches and reasonably structured surface meshes as the input to a dual creation algorithm.) However, the complexity of these algorithms often make incorporation of sheet quality criteria difficult.

More recent approaches, that take into account a better understanding of hexahedral constraints, offer a more realistic chance of success. The method proposed by Staten, et al. [55] inserts sheets directly based on boundary definitions building on some of the paradigms advanced in the plastering and H-morph [42] algorithms, but advancing entire sheets rather than single elements to avoid closure problems.

Indirect Methods

The Indirect Methods are methods that generate a hexahedral mesh by first generating a tetrahedral mesh and then converting that mesh into a hexahedral mesh by either combining the tetrahedra into hexahedra, or decomposing the tetrahedra into hexahedra. Combining tetrahedra into hexahedra is ultimately very similar to the plastering method, which we discussed previously.

Indirect Decomposition

The major advantage of tetrahedral decomposition into hexes that it is very easy to satisfy the boundary and topologic constraints of hexahedral meshes. The disadvantage of this method is that the decomposition produces sheets that are small and spherical, resulting in relatively high curvature of all sheets within the solid.

To augment an indirect decomposition algorithm, methods that reduce the curvature of each of the sheets in the volume will improve the quality of the existing mesh, and also reduce the high nodal valence that often accompany

a tetrahedral mesh. Some ideas for accomplishing this could include combining adjacent sheets into single sheets, which elongates the sheets reducing curvature and locally coarsening the mesh (i.e. combining two slightly offset spheres produces an elliptical sheet). Iterative and strategic combinations of sheets could ultimately produce a mesh whose sheets are much more planar in nature, resulting in a higher quality mesh.

Another approach to this problem is to begin with a very coarse tetrahedral mesh, split each tetrahedra into four hexahedra, and then apply the dicing algorithm to mesh (see [45, 44]). The resulting mesh has few very large spherical sheets whose local curvature is relatively low, and the sheets produced via dicing are copies of this sheet still maintaining the relatively low curvature of the large spherical sheet.

5 OPEN PROBLEMS

In the course of preparing this survey of hexahedral meshing constraints and evaluating existing algorithms based on these constraints, several questions arose that remain open problems in hexahedral mesh generation. We list some of these questions below:

- **Is it possible to develop an algorithm which rearranges mesh topology to improve geometric quality of the sheets?** Current smoothing algorithms modify the placement of nodes within a mesh in order to improve the quality of the mesh without altering the mesh topology (that is, smoothing algorithms do not change the connectivity of an element with adjacent elements). Mesh flipping algorithms exist which rearrange hexahedral mesh topology [3, 4, 58]. Typically, these algorithms perform a 'flip' operation, and then compare the quality of the meshes before and after the operation [21]. If the operation tends to reduce curvature of the sheet, for example, then several flip operations strategicall performed in succession may possibly improve the quality of the mesh beyond what is possible with normal smoothing operations. Is it possible to 'drive' these flip operations utilizing geometric information from the sheets to realize dramatic improvements in overall mesh quality?
- **Assuming all topologic constraints are satisfied within a mesh, does there always exist an untangled solution for that mesh?** Current mesh smoothing and optimization algorithms, including mesh untangling algorithms are based on optimization algorithms for finding local extrema. Assuming that all topologic criteria for a hexahedral mesh have been satisfied, what are the conditions which prevent mesh untangling? Is it possible to prove that only tangled solutions exist for some mesh topologies and geometries using global optimization algorithms? Assuming that a mesh with a tangled solution exists, is it possible to show that the tangling is caused by boundary effects? Can the boundary effects be mitigated with

additional algorithms which may change the mesh topology but improve the geometry of the sheets?

- **Is it possible to calculate ranges of potential element quality for a mesh given a mesh topology and a fixed boundary?** Or, in other words, given two mesh topologies for a given geometry, is it possible to show that one mesh topology is likely to have higher quality after optimization than the other mesh topology?
- **Is it possible to determine a minimal set of sheets necessary to hexahedrally mesh a given geometry?** Existing research into hexahedral refinement algorithms [60, 18] coupled with algorithms for generating coarse meshes, may offer an alternative method for hexahedral mesh generation. Utilizing a minimal set of sheets necessary for a given geometry, it may be possible to perform the complex calculations necessary to obtain an initial coarse mesh of a geometry, then utilize the refinement algorithms to obtain the final mesh with desired sizing characteristics. What is the fewest number of hexahedral elements necessary to decompose a given geometry?
- **Is it possible to extend any current hexahedral meshing algorithm to incorporate more implicitly constraint-satisfying methods to mesh a significantly larger collection of geometric models?** In the previous section, we discussed possible extensions to existing algorithms which could be incorporated to dramatically extend the class of geometries to which these algorithms apply. Extension of these algorithms remains an open area for development.
- **For a given hexahedrally meshed geometry, is it possible to identify elements contained within this mesh that will be present in any hexahedral mesh applied to that same geometry?** Assuming that each mesh for a given geometry must satisfy the hexahedral mesh constraints outlined in this paper, can we identify the elements that must be present in all meshes of that same geometry?

6 CONCLUSION

A thorough understanding of the constraints associated with hexahedral mesh generation may provide the insight necessary to dramatically reduce the time required to generate hexahedral meshes for use in numerical analyses. Due to the difficulty and complexity of generating hexahedral meshes, however, tetrahedral meshes are often the only practical means available when performing numerical analysis.

In this paper, we have outlined many of the necessary constraints for generating a hexahedral mesh as derived from the dual representation of a hexahedral mesh. The constraints were classified as topology, boundary, and geometric, or quality, constraints, where topology defines how sheets are allowed to interact with one another, boundary defines how sheets capture geometric

features of the geometry, and geometric, or quality, defines interactions and conformationsthat are necessary to avoid element inversions and maintain high hexahedral element quality.

By incorporating methods to satisfy ignored, or overlooked, constraints in existing hexahedral mesh generation algorithms, it is be possible to extend the class of geometries for which these algorithms may be applied. Several existing algorithms for satisfying individual constraints are also highlighted, and it is shown how these algorithms satisfy individual constraints to compatibly mesh a solid with hexahedral elements. We have also listed a series of open problems in relation to these constraints for which future research may provide valuable insights for hexahedral mesh generation.

References

1. ANSYS. Ansys icem cfd (available from http://www.ansys.com/products/icemcfd.asp), October 2006.
2. S. E. Benzley, E. Perry, K. Merkley, and B. Clark. A comparison of all hexagonal and all tetrahedral finite element meshes for elastic and elasto-plastic analysis. In *Proceedings, 4th International Meshing Roundtable*, pages 179–191. Sandia National Laboratories, October 1995.
3. M. Bern and D. Eppstein. Flipping cubical meshes. In *Proceedings, 10th International Meshing Roundtable*, pages 19–29. Sandia National Laboratories, October 2001.
4. M. Bern, D. Eppstein, and J. Erickson. Flipping cubical meshes. *Engineering with Computers*, 18(3):173–187, 2002.
5. T. D. Blacker. The cooper tool. In *Proceedings, 5th International Meshing Roundtable*, pages 13–30. Sandia National Laboratories, October 1996.
6. T. D. Blacker and R. J. Meyers. Seams and wedges in plastering: A 3d hexahedral mesh generation algorithm. *Engineering With Computers*, 2(9):83–93, 1993.
7. T. D. Blacker, J. L. Mitchiner, L. R. Phillips, and Y. Lin. Knowledge system approach to automated two-dimensional quadrilateral mesh generation. *Computers in Engineering*, 3:153–162, 1988.
8. M. J. Borden, S. E. Benzley, and J. F. Shepherd. Coarsening and sheet extraction for all-hexahedral meshes. In *Proceedings, 11th International Meshing Roundtable*, pages 147–152. Sandia National Laboratories, September 2002.
9. M. J. Borden, J. F. Shepherd, and S. E. Benzley. Mesh cutting: Fitting simple all-hexahedral meshes to complex geometries. In *Proceedings, 8th International Society of Grid Generation Conference*, 2002.
10. M. L. Bussler and A. Ramesh. The eight-node hexahedral elements in fea of part designs. *Foundry Management and Technology*, pages 26–28, November 1993.
11. N. A. Calvo and S. R. Idelsohn. All-hexahedral element meshing: Generation of the dual mesh by recurrent subdivision. *Computer Methods in Applied Mechanics and Engineering*, 182:371–378, 2000.
12. A. O. Cifuentes and A. Kalbag. A performance study of tetrahedral and hexahedral elements in 3-d finite element structural analysis. *Finite Elements in Analysis and Design*, 12(3-4):313–318, 1992.

13. W. A. Cook and W. R. Oakes. Mapping methods for generating three-dimensional meshes. *Computers In Mechanical Engineering*, CIME Research Supplement:67–72, August 1982.
14. J. F. Dannenhoffer(III). A block-structuring technique for general geometries. In *29th Aerospace Sciences Meeting and Exhibit*, volume AIAA-91-0145, January 1991.
15. D. Eppstein. Linear complexity hexahedral mesh generation. In *12th ACM Symposium on Computational Geometry*, pages 58–67. ACM, 1996.
16. N. T. Folwell and S. A. Mitchell. Reliable whisker weaving via curve contraction. *Engineering With Computers*, 15:292–302, 1999.
17. L. A. Freitag and P. Plassmann. Local optimization-based simplicial mesh untangling and improvement. *International Journal for Numerical Methods in Engineering*, 49(1):109–125, September 10-20, 2000.
18. N. Harris, S. E. Benzley, and S. J. Owen. Conformal refinement of all-hexahedral meshes based on multiple twist plane insertion. In *Proceedings, 13th International Meshing Roundtable*, pages 157–168. Sandia National Laboratories, September 2004.
19. S. R. Jankovich, S. E. Benzley, J. F. Shepherd, and S. A. Mitchell. The graft tool: An all-hexahedral transition algorithm for creating multi-directional swept volume mesh. In *Proceedings, 8th International Meshing Roundtable*, pages 387–392. Sandia National Laboratories, October 1999.
20. P. Knupp. Next-generation sweep tool: A method for generating all-hex meshes on two-and-one-half dimensional geometries. In *Proceedings, 7th International Meshing Roundtable*, pages 505–513. Sandia National Laboratories, October 1998.
21. P. Knupp and S. A. Mitchell. Integration of mesh optimization with 3d all-hex mesh generation, ldrd subcase 3504340000, final report. SAND 99-2852, October 1999.
22. P. M. Knupp. Algebraic mesh quality metrics. *SIAM J. Sci. Comput.*, 23(1):193–218, 2001.
23. P. M. Knupp. Hexahedral mesh untangling and algebraic mesh quality metrics. In *Proceedings, 9th International Meshing Roundtable*, pages 173–183. Sandia National Laboratories, October 2000.
24. J. Kraftcheck. *Virtual Geometry: A Mechanism for Modification of CAD Model Topology for Improved Meshability.* Published Master's Thesis, Department of Mechanical Engineering, University of Wisconsin, December 2000.
25. M. Lai, S. E. Benzley, G. D. Sjaardema, and T. J. Tautges. A multiple source and target sweeping method for generating all-hexahedral finite element meshes. In *Proceedings, 5th International Meshing Roundtable*, pages 217–228. Sandia National Laboratories, October 1996.
26. M. Lai, S. E. Benzley, and D. R. White. Automated hexahedral mesh generation by generalized multiple source to multiple target sweeping. *International Journal for Numerical Methods in Engineering*, 49(1):261–275, September 2000.
27. M. Loriot. Tetmesh-ghs3d v3.1 the fast, reliable, high quality tetrahedral mesh generator and optimiser (see http://www.simulog.fr/mesh/tetmesh3p1d-wp.pdf).
28. Y. Lu, R. Gadh, and T. J. Tautges. Volume decomposition and feature recognition for hexahedral mesh generation. In *Proceedings, 8th International Meshing Roundtable*, pages 269–280. Sandia National Laboratories, October 1999.

29. S. Means, A. J. Smith, J. F. Shepherd, J. Shadid, J. Fowler, R. Wojcikiewicz, T. Mazel, G. D. Smith, and B. S. Wilson. Impact of geometry on spatial distributions of intralumenal endoplasmic reticulum calcium under variant ip3r channel distributions. *to appear in Biophys. J.*, accepted March 2006.
30. D. L. Meier. Multidimensional astrophysical structural and dynamical analysis. i. development of a nonlinear finite element approach. *Astrophys. J.*, 518:788–813, 1999.
31. D. J. Melander. *Generation of Multi-Million Element Meshes for Solid Model-Based Geometries: The Dicer Algorithm.* Published Master's Thesis, Brigham Young University, April 1997.
32. D. J. Melander, T. J. Tautges, and S. E. Benzley. Generation of multi-million element meshes for solid model-based geometries: The dicer algorithm. *AMD - Trends in Unstructured Mesh Generation*, 220:131–135, July 1997.
33. K. A. Milton. Finite-element quantum field theory. In *Proceedings of the XIVth International Symposium on Lattice Field Theory*, volume Nucl. Phys. B(Proc. Suppl.) 53 (1997), pages 847–849, 1996.
34. S. A. Mitchell. A characterization of the quadrilateral meshes of a surface which admit a compatible hexahedral mesh of the enclosed volumes. In *13th Annual Symposium on Theoretical Aspects of Computer Science*, volume Lecture Notes in Computer Science: 1046, pages 465–476, 1996.
35. S. A. Mitchell. Choosing corners of rectangles for mapped meshing. In *13th Annual Symposium on Computational Geometry*, pages 87–93. ACM Press, June 1997.
36. S. A. Mitchell. High fidelity interval assignment. In *Proceedings, 6th International Meshing Roundtable*, pages 33–44. Sandia National Laboratories, October 1997.
37. S. A. Mitchell and T. J. Tautges. Pillowing doublets: Refining a mesh to ensure that faces share at most one edge. In *Proceedings, 4th International Meshing Roundtable*, pages 231–240. Sandia National Laboratories, October 1995.
38. M. Muller-Hannemann. Hexahedral mesh generation by successive dual cycle elimination. In *Proceedings, 7th International Meshing Roundtable*, pages 365–378. Sandia National Laboratories, October 1998.
39. P. J. Murdoch. *The Spatial Twist Continuum: A Dual Representation of the All Hexahedral Finite Element Mesh.* Published Doctoral Dissertation, Brigham Young University, December 1995.
40. P. J. Murdoch and S. E. Benzley. The spatial twist continuum. In *Proceedings, 4th International Meshing Roundtable*, pages 243–251. Sandia National Laboratories, October 1995.
41. S. J. Owen. A survey of unstructured mesh generation technology (available at http://www.andrew.cmu.edu/user/sowen/survey/index.html), September 1998.
42. S. J. Owen and S. Saigal. H-morph an indirect approach to advancing front hex meshing. *International Journal for Numerical Methods in Engineering*, 49(1):289–312, September 2000.
43. Pointwise. Gridgen - reliable cfd meshing (available from http://www.pointwise.com/gridgen/), October 2006.
44. S. Richards, S. E. Benzley, J. F. Shepherd, and M. B. Stephenson. Dthexing: An improved all-hexahedral meshing scheme using general coarsening tools. SAND2003-2724A and SAND2003-2818P, 2003.

45. S. Richards, S. E. Benzley, J. F. Shepherd, and M. B. Stephenson. Dthexing: Creation of semi-structured all-hexahedral meshes based on coarse tetrahedral primitives. Sandia doc. num. 5237908, November 2005.
46. X. Roca, J. Sarrate, and A. Huerta. Surface mesh projection for hexahedral mesh generation by sweeping. In *Proceedings, 13th International Meshing Roundtable*, volume SAND 2004-3765C, pages 169–180. Sandia National Laboratories, September 2004.
47. R. Schneiders. An algorithm for the generation of hexahedral element meshes based on an octree technique. In *Proceedings, 6th International Meshing Roundtable*, pages 183–194. Sandia National Laboratories, October 1997.
48. Schneiders Pyramid Open Problem, http://www-users.informatik.rwth-aachen.de/ roberts/open.html.
49. A. Schwartz and G. M. Ziegler. Construction techniques for cubical complexes, odd cubical 4-polytopes, and prescribed dual manifolds. *Experiment. Math.*, 13(4):385–413, 2004.
50. M. S. Shephard and M. K. Georges. Three-dimensional mesh generation by finite octree technique. *International Journal for Numerical Methods in Engineering*, 32:709–749, 1991.
51. J. F. Shepherd, S. E. Benzley, and S. A. Mitchell. Interval assignment for volumes with holes. *International Journal for Numerical Methods in Engineering*, 49(1):277–288, September 2000.
52. J. F. Shepherd, S. A. Mitchell, P. Knupp, and D. R. White. Methods for multisweep automation. In *Proceedings, 9th International Meshing Roundtable*, pages 77–87. Sandia National Laboratories, October 2000.
53. J. R. Shewchuk. What is a good linear element? interpolation, conditioning, and quality measures. In *Proceedings, 11th International Meshing Roundtable*, pages 115–126. Sandia National Laboratories, September 2002.
54. M. L. Staten, S. A. Canann, and S. J. Owen. Bmsweep: Locating interior nodes during sweeping. In *Proceedings, 7th International Meshing Roundtable*, pages 7–18. Sandia National Laboratories, October 1998.
55. M. L. Staten, S. J. Owen, and T. D. Blacker. Unconstrained paving and plastering: A new idea for all hexahedral mesh generation. In *Proceedings, 14th International Meshing Roundtable*, pages 399–416. Sandia National Laboratories, September 2005.
56. T. Suzuki, S. Takahashi, and J. F. Shepherd. Practical interior surface generation method for all-hexahedral meshing. In *Proceedings, 14th International Meshing Roundtable*, pages 377–397. Sandia National Laboratories, September 2005.
57. T. J. Tautges, T. D. Blacker, and S. A. Mitchell. Whisker weaving: A connectivity-based method for constructing all-hexahedral finite element meshes. *International Journal for Numerical Methods in Engineering*, 39:3327–3349, 1996.
58. T. J. Tautges and S. Knoop. Topology modification of hexahedral meshes using atomic dual-based operations. In *Proceedings, 12th International Meshing Roundtable*, pages 415–423. Sandia National Laboratories, September 2003.
59. T. J. Tautges, S. sheng Liu, Y. Lu, J. Kraftcheck, and R. Gadh. Feature recognition applications in mesh generation. *AMD-Trends in Unstructured Mesh Generation*, 220:117–121, July 1997.
60. K.-F. Tchon, J. Dompierre, and R. Camarero. Conformal refinement of all-quadrilateral and all-hexahedral meshes according to an anisotropic metric. In

Proceedings, 11th International Meshing Roundtable, pages 231–242. Sandia National Laboratories, September 2002.
61. B. Thurston. Geometry in action: Hexahedral decomposition of polyhedra (available from http://www.ics.uci.edu/ eppstein/gina/thurston-hexahedra.html), October 1993.
62. P. Vachal, R. V. Garimella, and M. J. Shashkov. Mesh untangling. LAU-UR-02-7271, T-7 Summer Report 2002.
63. K. S. Walton, S. E. Benzley, and J. F. Shepherd. Sculpting: An improved inside-out scheme for all-hexahedral meshing. In *Proceedings, 11th International Meshing Roundtable*, pages 153–159. Sandia National Laboratories, September 2002.
64. F. R. S. Weiler and R. Schneiders. Automatic geometry-adaptive generation of quadrilateral and hexahedral element meshes for fem. In *Proceedings, 5th International Conference on Numerical Grid Generation in Computational Field Simulations*, pages 689–697. Mississippi State University, April 1996.
65. V. I. Weingarten. The controversy over hex or tet meshing. *Machine Design*, pages 74–78, April 18, 1994.
66. D. R. White. *Automatic Quadrilateral and Hexahedral Meshing of Pseudo-Cartesian Geometries Using Virtual Subdivision.* Published Master's Thesis, Brigham Young University, June 1996.
67. D. R. White, M. Lai, S. E. Benzley, and G. D. Sjaardema. Automated hexahedral mesh generation by virtual decomposition. In *Proceedings, 4th International Meshing Roundtable*, pages 165–176. Sandia National Laboratories, October 1995.
68. D. R. White, R. W. Leland, S. Saigal, and S. J. Owen. The meshing complexity of a solid: An introduction. In *Proceedings, 10th International Meshing Roundtable*, pages 373–384. Sandia National Laboratories, October 2001.
69. D. R. White, S. Saigal, and S. J. Owen. Meshing complexity of single part cad models. In *Proceedings, 12th International Meshing Roundtable*, pages 121–134. Sandia National Laboratories, September 2003.
70. D. R. White and T. J. Tautges. Automatic scheme selection for toolkit hex meshing. *International Journal for Numerical Methods in Engineering*, 49(1):127–144, September 2000.
71. M. Whitely, D. R. White, S. E. Benzley, and T. D. Blacker. Two and three-quarter dimensional meshing facilitators. *Engineering With Computers*, 12:155–167, 1996.
72. I. XYZ Scientific Applications. Truegrid: High quality hexahedral grid and mesh generation for fluids and structures (available from http://www.truegrid.com), October 2006.
73. M. A. Yerry and M. S. Shephard. Three-dimensional mesh generation by modified octree technique. *International Journal for Numerical Methods in Engineering*, 20:1965–1990, 1984.
74. Y. Zhang and C. Bajaj. Adaptive and quality quadrilateral/hexahedral meshing from volumetric imaging data. In *Proceedings, 13th International Meshing Roundtable*, pages 365–376. Sandia National Laboratories, September 2005.
75. Y. Zhang, C. Bajaj, and G. Xu. Surface smoothing and quality improvement of quadrilateral/hexahedral meshes with geometric flow. In *Proceedings, 14th International Meshing Roundtable*, pages 449–468. Sandia National Laboratories, September 2005.